View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Washington University St. Louis: Open Scholarship

Washington University in St. Louis

Washington University Open Scholarship

All Computer Science and Engineering Research

Computer Science and Engineering

Report Number: WUCS-90-29

1990-09-01

Studies in the Hybrid Deterministic Parsing

Stan C. Kwasny, Anne M. Johnstone, and Barry L. Kalman

This report details research plans to extend work on hybrid deterministic parsing. The approach is hybrid in its combination of symbolic and sub-symbolic (connectionist) approaches. The research is investigating (1) new and more robust architectures for deterministic parsing which are hybrid mixtures of symbolic and sub-symbolic components, (2) combinations of syntax with lexical access and other components of natural language processing in order to determine how the distributed patterns of connectionism may enable better interfaces among these components, and (3) properties of the rules in the rule-based deterministic processing of natural language that make them most suitable for sub-symbolic... **Read complete abstract on page 2.**

Follow this and additional works at: https://openscholarship.wustl.edu/cse_research

 Part of the Computer Engineering Commons, and the Computer Sciences Commons

Recommended Citation

Kwasny, Stan C.; Johnstone, Anne M.; and Kalman, Barry L., "Studies in the Hybrid Deterministic Parsing" Report Number: WUCS-90-29 (1990). *All Computer Science and Engineering Research.*
https://openscholarship.wustl.edu/cse_research/704

Department of Computer Science & Engineering - Washington University in St. Louis
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160.

This technical report is available at Washington University Open Scholarship: https://openscholarship.wustl.edu/cse_research/704

Studies in the Hybrid Deterministic Parsing

Stan C. Kwasny, Anne M. Johnstone, and Barry L. Kalman

Complete Abstract:

This report details research plans to extend work on hybrid deterministic parsing. The approach is hybrid in its combination of symbolic and sub-symbolic (connectionist) approaches. The research is investigating (1) new and more robust architectures for deterministic parsing which are hybrid mixtures of symbolic and sub-symbolic components, (2) combinations of syntax with lexical access and other components of natural language processing in order to determine how the distributed patterns of connectionism may enable better interfaces among these components, and (3) properties of the rules in the rule-based deterministic processing of natural language that make them most suitable for sub-symbolic training. Certainly, both symbolic and sub-symbolic components are important for language processing. This research is attempting to understand how best to make the division between the two types of components. Our goal is to provide insights from his perspective and to show that the two approaches, if appropriately combined, can mutually benefit each other. Furthermore, the development of representation techniques which encode ambiguity (rather than making arbitrary choices) is being examined as a useful methodological shift in dictating how systems can be decomposed into their parts. Various system architectures are being explored over a spectrum of symbolic and sub-symbolic arrangements. We are exploring the proper connections among the parts while integrating the components required for NLP. These systems are organized around syntax, but not exclusively limited to it. We want to achieve a better understanding of such systems and attain a clearer idea of which components are best accomplished symbolically and which ones sub-symbolically.

STUDIES IN
HYBRID DETERMINISTIC PARSING

Stan C. Kwasny
Anne M. Johnstone
Barry L. Kalman

WUCS-90-29

September 1990

Center for Intelligent Computing Systems
Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
St. Louis, Missouri 63130-4899
sck@wucs1.wustl.edu
amj@wucs1.wustl.edu
barry@wucs1.wustl.edu

The sponsors of the Center are McDonnell Douglas Corporation and Southwestern Bell Telephone Company.

STUDIES IN HYBRID DETERMINISTIC PARSING

Stan C. Kwasny, Anne M. Johnstone, Barry L. Kalman

WUCS-90-29

Center for Intelligent Computing Systems
Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
St. Louis, Missouri 63130-4899
sck@wucs1.wustl.edu, amj@wucs1.wustl.edu, barry@wucs1.wustl.edu

Abstract

This report details research plans to extend work on hybrid deterministic parsing. The approach is hybrid in its combination of symbolic and sub-symbolic (connectionist) approaches. The research is investigating (1) new and more robust architectures for deterministic parsing which are hybrid mixtures of symbolic and sub-symbolic components, (2) combinations of syntax with lexical access and other components of natural language processing in order to determine how the distributed patterns of connectionism may enable better interfaces among these components, and (3) properties of the rules in the rule-based deterministic processing of natural language that make them most suitable for sub-symbolic training.

Certainly, both symbolic and sub-symbolic components are important for language processing. This research is attempting to understand how best to make the division between the two types of components. Our goal is to provide insights from this perspective and to show that the two approaches, if appropriately combined, can mutually benefit each other. Furthermore, the development of representation techniques which encode ambiguity (rather than making arbitrary choices) is being examined as a useful methodological shift in dictating how systems can be decomposed into their parts. Various system architectures are being explored over a spectrum of symbolic and sub-symbolic arrangements. We are exploring the proper connections among the parts while integrating the components required for NLP. These systems are organized around syntax, but not exclusively limited to it. We want to achieve a better understanding of such systems and attain a clearer idea of which components are best accomplished symbolically and which ones sub-symbolically.

Table of Contents

1. Introduction

Natural Language Processing (NLP) by computer is elusive. How can native English speakers process and generate English so adeptly while computational and other models that look so promising in theory yield such limited results?

While such pessimism may not be widespread among Natural Language researchers, it is shared by many. For example, Sondheimer [51] questions why progress has been so slow in the field. In fact, Winograd and Flores [58] believe that "*..computers cannot understand language*" (p.107). At the recent international conference for Computational Linguistics in Helsinki (COLING-90), organizers reserved time to discuss "the unfinished language." This at least concedes that the task of NLP by computer is not complete and much is left to be done before the problems are solved. These are embarrassing, but important questions to ask especially in discussing work on Natural Language Processing.

Realistically, the task of processing and understanding Natural Language is very complex. It should be obvious, therefore, that solutions, even partial solutions, must be capable of addressing that complexity through the proper abstraction mechanisms. The task must necessarily be divided into smaller pieces in order to have any hope of making progress, but dividing up the task incorrectly will not generally lead to progress with any lasting character. And yet some simplification is necessary since no single theory yet proposed, nor even imaginable, encompasses all of human thought. Such a theory of human thought would be necessary if a single complete account of language is to be provided. In simplifying the problem, however, care must be taken not to oversimplify, nor to trivialize important components. The results should scale to further dimensions as more of the solution unfolds. Many promising approaches have had difficulty generalizing to new domains and scaling to account for more of the problem.

Connectionism provides a perspective on the decomposition of large tasks into smaller ones. Similar to the blackboard architectures [13] that grew out of research on speech, various modules (called knowledge sources in blackboard systems) can often provide a degree of expertise, but are unable to fully address the larger task alone. Distributed representations provide a means by which partial solutions can be encoded as activation patterns leaving the remainder of the solution to other components. Here, the notion of competition is fundamental. While a particular part of the solution is unknown, possible solutions compete based on corroborative evidence from relevant sources.

Work in natural language processing over the past few decades by both psycholinguists (e.g., Marslen-Wilson [35]; Marslen-Wilson and Tyler [36]; Cole and Jakimik [8]; Grosjean [21]), and cognitive scientists (e.g., McClelland and Elman [37]; Cottrell [9]) has shown the difficulties of concentrating too exclusively on any one aspect of a natural language system. Cottrell [9] laments that he started out in his work investigating lexical ambiguity, but ended up with a full model of sentence comprehension since he needed to specify the sources of the disambiguating information.

One approach, to be further examined in this work, comes from the choice of representation for language structures. Ambiguities that inevitably occur in language can produce a combinatorial number of processing choices. This is clearly seen, for example, in speech processing. Psycholinguistic studies have shown that people use as much information as possible, as quickly as possible, from many different sources, when interpreting an utterance. We argue that such processing is necessary and not just facilitative. In recent studies of a large-scale speech recognition system [22,26] it is argued that without the immediate constraints of semantic and other knowledge a combinatorial explosion of partial hypotheses results and rapidly grows unmanageable.

Many problems are not apparent unless one studies the dynamics of a system, and the problems the output from one component may pose for another. In the speech domain for example, just a few lexical competitors over the same stretch of sound can quickly grow combinatorially into a huge number of word

strings when viewed from the syntactic level. Syntactic and semantic information can (at least initially) actually increase the confusion by adding multiple syntactic categories and multiple meanings for each word. Cottrell cites the potential proliferation of partial semantic interpretations generated by the scores of different meanings associated with such every day words as "run." For these reasons we believe that what goes on *between* the components of a system is at least as important as what goes on *within* a component.

In the symbolic processing of Natural Language, similar effects can be found. Small and Rieger [1,50] have addressed the problem of NLP as a matter of combining word senses into sentence structures. The laborious effort of building the lexicon to capture all subtle meaning differences of all possible words suffers from the same combinatorial problems and misses important generalizations. The answer to this question that we wish to investigate hinges on the ability to build representations which carry the ambiguities as part of the structure itself. Distributed representations can be adapted for this purpose. They feature a more compact way to represent sets of choices than the discrete world of symbolic structures.

Connectionist approaches to Natural Language Processing, while by most accounts not yet as successful as symbolic approaches, stand as an important counterpoint to current approaches. Connectionism features robust decision making, generalization, and other benefits of an extensional style of programming, while symbolic approaches enjoy a history of linguistic tradition, the application of well-understood methods, and the security that comes from computing intermediate structures to reassure that processing is proceeding according to design.

In 1987, the TINLAP-3 [57] panel on connectionist and other parallel approaches to NLP were asked the following questions:

> *"Is Natural Language Processing inevitably committed to a symbolic form of representation? Can syntactic, semantic, or world knowledge be represented in that paradigm if taken seriously? What parts of current Computational Linguistics will fare worst if there turn out to be significant empirical advances with connectionist parsing? Are there any yet (i.e., how far do we trust simulations programmed only on serial machines)?*
>
> *"What new approaches to syntax, semantics or pragmatics will be needed if this approach turns out to be empirically justified? Will it just bring back all the old views associated with associationism, and will they be changed in the journey? Is parallel parsing just a new implementation or a real paradigm shift?"*

While answers to most of these questions were inconclusive at the time, it is clear that some discontent is in evidence and that progress is not as rapid as one would like within the symbolic paradigm alone. To date, connectionist parsers have exhibited encouraging results but have not matured to the point of tackling many of the more challenging problems.

Language is the crucial test case as seen by Pinker and Prince [43] who state:

> *"Language has been the domain most demanding of articulated symbol structures governed by rules and principles and it is also the domain where such structures have been explored in the greatest depth and sophistication, within a range of theoretical frameworks and architectures, attaining a wide variety of significant empirical results. Any alternative model that either eschews symbolic mechanisms altogether, or that is strongly shaped by the restrictive nature of available elementary information processes and unresponsive to the demands of the high-level functions being computed, starts off at a seeming disadvantage. Many observers thus feel that connectionism, as a radical restructuring of cognitive theory, will stand or fall depending on its ability to account for human language."* (p.78)

The processing of Natural Language is, at the same time, naturally symbolic and naturally subsymbolic, i.e., below the level of symbols. As Pinker and Prince point out, the symbolic viewpoint has predominated since in many respects symbols play a critical role. Writing systems, as one example, owe their existence to the symbolic nature of language. There is also a major component of language that is

not symbolic in this sense. Indeed, language must also be sub-symbolic because of the nature of speech, the fuzziness of concepts, and the high degree of parallelism that is difficult to explain as a purely symbolic phenomenon. Building a processor of Natural Language, therefore, should require a hybrid approach — one that combines both symbolic, sequential processing and sub-symbolic, parallel processing.

Clearly there are benefits in both symbolic and sub-symbolic approaches. Just as clearly there are drawbacks. Symbolic approaches tend to be brittle and intolerant to minor variations. They often contain rules which are difficult to compose or learn symbolically. Sub-symbolic NLP is still an exploratory endeavor which generally cuts against conventional wisdom by not supporting stacks and other hierarchical data structures. Such models traditionally impose critical limitations, such as arbitrarily limiting the maximum lengths of their input sentences or similarly limiting the iterative nature of the processing. Predictive experiments, for example using Elmannets [12], do not show how to process language in the traditional sense.

Our research combines the traditional symbolic approach with one based on connectionism and investigates ways of maintaining the benefits on both sides as much as possible while minimizing their disadvantages. We are exploring the proper connections while integrating the components required for NLP. The system is organized around syntax, but not exclusively limited to it. In the process, we are attempting to achieve a better understanding of such systems and to attain a clearer idea of which components are best accomplished symbolically and which ones sub-symbolically.

2. Background

Before describing our goals in this research in detail, some background material will be summarized. Here, we will concentrate on matters that relate directly to this report. Briefly, we define deterministic language processing. We next describe our approach in building a hybrid deterministic NLP system. Finally, we briefly identify some important areas of neural network research which have impacted our work.

2.1. Deterministic Processing of Language

In our work, we are assuming the validity of the determinism hypothesis [34] that Marcus put forth in his work on PARSIFAL. It states, in its revised form, that

> "*There is enough information in the structure of natural language in general, and in English in particular, to allow left-to-right deterministic parsing of those sentences which a native speaker can analyze without conscious effort.*" (p. 204)

With the development of PARSIFAL, the determinism hypothesis became a viable theory of language processing. It showed that a rule-based system could be developed to syntactically process a large subset of a natural language without backtracking and without building unnecessary structure. Furthermore, it suggested a correspondence between the language subset that could be processed in this manner and the language processing humans perform below the level of conscious effort. It thereby omits those "garden-path" sentences that apparently require some amount of re-analysis at a conscious level by humans.

As Marcus stated, deterministic parsers process input sentences primarily left-to-right. Determinism is accomplished by permitting a lookahead of up to three constituents with a constituent buffer designated for that purpose. To permit embedded structures, a stack is also part of the architecture. (Waltz and Pollack [53] describe this mechanism as one based on *delay*.)

Rules are partitioned into rule sets and a single processing step consists of selecting a rule, firing the rule, and performing its actions. Rule sets are usually associated with the current (top-level) node of the structure being built. Conflicts within the rule set are resolved from the static ordering (priority) of rules within each rule set and through special diagnostic rules. Actions effect changes to the stack and buffer. After a series of processing steps, a termination rule fires and processing is halted leaving the final structure on top of the stack.

2.2. Connectionist Processing of Language

Several researchers have attempted to process natural language using connectionism. In many of these parsers (e.g., Cottrell [9], Fanty [17], or Waltz and Pollack [53]) grammar rules are processed into a network of units connected with excitatory and inhibitory links. The number of units required to realize a given grammar is a function of the maximum input sentence length and the complexity of the grammar. Hence, a limitation is introduced on the number of elements that can be present in the input. Sentences are processed within such a framework by presenting them, possibly in a simulated left-to-right fashion, at the input side of the network and activations are permitted to spread through the network. Other approaches such as simulated annealing [48] have also been attempted with limited success.

Any plausible model of language processing should permit alternative linguistic structures to compete (in the same sense as the TRACE [37] model) while inputs are processed left-to-right. Computer models based on backtracking (e.g., Augmented Transition Networks (ATNs) or Definite Clause Grammars (DCGs)) do not adequately capture the competitive nature of sentence processing since there is no evidence from human experiments that any conscious re-processing of inputs is routinely performed. The lone exception seems to be for "garden path" sentences.

Connectionist methods for constructing parsers which process language naturally on an element-by-element basis have been investigated by Elman [12]. In his architecture, recurrent connections are permitted that link the hidden layer of the network to context units in the input layer. When compared to networks studied by Jordan [27] whose recurrent links originate from the output layer, these networks are thought to be more powerful [10]. The recurrent links provide information from the previous state of the process. It is important that this information be available at the current decision-making point so that parsing decisions can depend on what has been seen previously. Elman's experiments are not designed to represent structure. Next-word predictions are used to demonstrate the nature of grammatical knowledge learned. Similarly, Gigley's HOPE system [20] has proven effective in achieving predictive capabilities and in modeling certain aspects of aphasia.

In classic approaches, natural language processing by computer is performed under the direction of a set of grammar rules. These are often executed as if following instructions in a program. (In fact, upon examining the rules of PARSIFAL, one would be hard-pressed to distinguish PIDGIN, Marcus' English-like specification language, from a standard programming language.) If we are modeling human sentence processing, then this method is incorrect. Rules should be permitted to play an advisory role only — that is, as descriptions of typical situations and not as prescriptions for precise processing. Control in the application of a rule or variant of a rule should be determined jointly as a data-driven and expectation-driven process.

Symbolic rules are an essential part of most linguistic accounts at virtually all levels of processing, from speech signal to semantics. But systems based too literally on rules tend to be brittle since there is no direct way to process linguistic forms that do not strictly adhere to the pre-conceived rules. If a complete set of rules for all meaningful English forms existed, then this might be satisfactory. But no such set of rules exists, nor does it seem desirable or even possible to construct such a set.

Another consequence of a rule-based grammar is a pragmatic one. As with any rule-based system, the acquisition of new grammar rules often require tedious re-tuning of existing rules. Rarely can a rule be added to the grammar without it affecting and being affected by other rules in the grammar. To the credit of their creators, some grammars have been continually refined over a period of years, even decades, in an attempt to more accurately depict the processing requirements of English. If rules are accepted as important in the process, then the only solution to this problem in a practical and realistic manner is through learning.

Our hybrid parser [29] is composed of a connectionist network trained from rule ''templates'' or sentence traces which are derived from a deterministic grammar. Actions in the hybrid parser are performed symbolically on traditional data structures which are also maintained symbolically. The symbolic component manages the input sentence and the flow of constituents into the lookahead buffer, coding them as required for the input level of the network in the sub-symbolic component. On the return side, it evaluates the activations of the output units, decides which action to perform, and performs that action, potentially modifying the stack and buffer in the process.

The responsibility of the sub-symbolic component, therefore, is to examine the contents of the buffer and stack and yield a preference for a specific action. Learning itself occurs off-line and can be a time-consuming process, but once learned the processing times for the system are excellent. Computations need only flow in one direction in the network. The feed-forward multiplication of weights and computation of activation levels for individual units produce the pattern of activation on the output level. Activation of output units is interpreted in a winner-take-all manner, so that the unit with the highest activation determines the action to be taken.

Training of our hybrid parser proceeds by presenting patterns to a layered network and teaching it to respond with an appropriate action using backward error propagation [47,56]. The input patterns represent encodings of the buffer positions and the top of the stack from the deterministic parser. Note that this is of fixed-size. The output of the network contains a series of units each representing an action to be performed during processing and judged in a winner-take-all fashion. Note also that the number of actions is also limited according to the specification of the grammar. Network convergence is observed once the network can achieve a perfect score on the training patterns themselves and the error measure has decreased to an acceptable level, determined empirically.

A sentence form is parsed by iteratively presenting the network with inputs coded from the stack and buffer and performing the action designated by the network. The action modifies the stack and buffer and the process repeats.

Each sentence receives a score representing the overall average strength of responses during processing. The score for each processing step is computed as the reciprocal of the error for that step. The error is computed as the Euclidean distance between the actual output and an idealized output consisting of a **-1** value for every output unit except the winning unit which has a **+1** value. The errors for each step are summed and averaged over the number of steps. The average strength is the reciprocal of the average error per step.

In [16] we have examined two distinct approaches to training a network to parse sentence forms. Each of these training strategies result in a slightly different version of the hybrid parser. The difference lies in the nature of the training patterns presented. One approach uses rule templates, training patterns derived from the rules. This type of learning is deductive in the sense that a very general form of each rule is learned from which the parser must derive actions specific to individual cases. The second approach uses training data derived from sentence processing traces. This form of training is inductive in the sense that the parser must arrive at general patterns of performance from the specific instances

presented. The goal of both deductive and inductive training is to produce a network capable of mimicking the rules or sentence forms on which its training is based and to do so in a way that generalizes to many additional cases. Once initial learning has been accomplished, simulation experiments can be performed to examine the generalization capabilities of the resulting networks.

In deductive training, each grammar rule is coded as a training template which is a list of feature values, but templates are not grouped into rule packets. In general, each constituent is represented by an ordered feature vector in which one or more values is ON(**+1**) for features of the form and all other values are either OFF(**-1**) or DO NOT CARE (**?**). A rule template is instantiated by randomly changing **?** to **+1** or **-1**. Thus, each template represents many training patterns and each training epoch is slightly different. During training, the network learns the inputs upon which it can rely in constructing its answers and which inputs it can ignore.

The probability of a **?** becoming a **+1** or **-1** varies according to the training that we are conducting. Each rule template containing n **?**'s can generate up to 2^n unique training cases. Some rule templates have over 30 **?**'s which means they represent approximately 10^9 training cases. It is obviously impossible to test the performance of all these cases, so each **?** in the rule template is replaced by an estimate of the expected value of an input unit to provide testing patterns. While in actual processing such an activation level for a feature may never be encountered, it is a good test value since it represents an estimate of the mean of the range of values seen during training.

Grammar rules are coded into rule templates by concatenating the feature vectors of the component constituents from the stack and buffer. Each grammar rule takes the following form:

{ <Stack> <1st Item> <2nd Item> <3rd Item> → Action }

For example, a rule for Yes/No questions, modeled after PARSIFAL, could be written:

{ < *S node* > < *"have"* > < *NP* > < *VERB, -en* > → *Switch 1st and 2nd items* }

while a rule for imperative sentences could be written:

{ < *S node* > < *"have"* > < *NP* > < *VERB, inf* > → *Insert YOU* }

By replacing each constituent with its coding, a rule template is created. In the two rules above, rule templates are created with a **?** value for many of the specific verb features of the initial form "have" in each rule, but are carefully coded in the third buffer position where the primary differences lie. Because different actions are required, these are also coded to have different target values during training.

A second type of training for the network uses training patterns derived from traces of the situations encountered and actions performed during the rule-based processing of actual sentences. This processing is guided by application of the rules of a deterministic grammar as before. The rules need not actually be developed nor written down in advance as long as the contents of buffer and stack can be associated with an action. Figure 1 compares deductive and inductive strategies in terms of how the training patterns are extracted.

With deductive training, each pattern template often represents an enormous number of training patterns. For this reason, convergence is not simply a matter of computing the total sum of squares or similar error value from an epoch of training patterns. In our method, an epoch is one pass through all templates, but each epoch, therefore, is different. Convergence cannot be defined in the conventional way. We can only test the performance of the network on expected, grammatical sentence forms. If it fails to correctly process a grammatical form, then our conclusion is that the network is not yet trained and more training is performed.

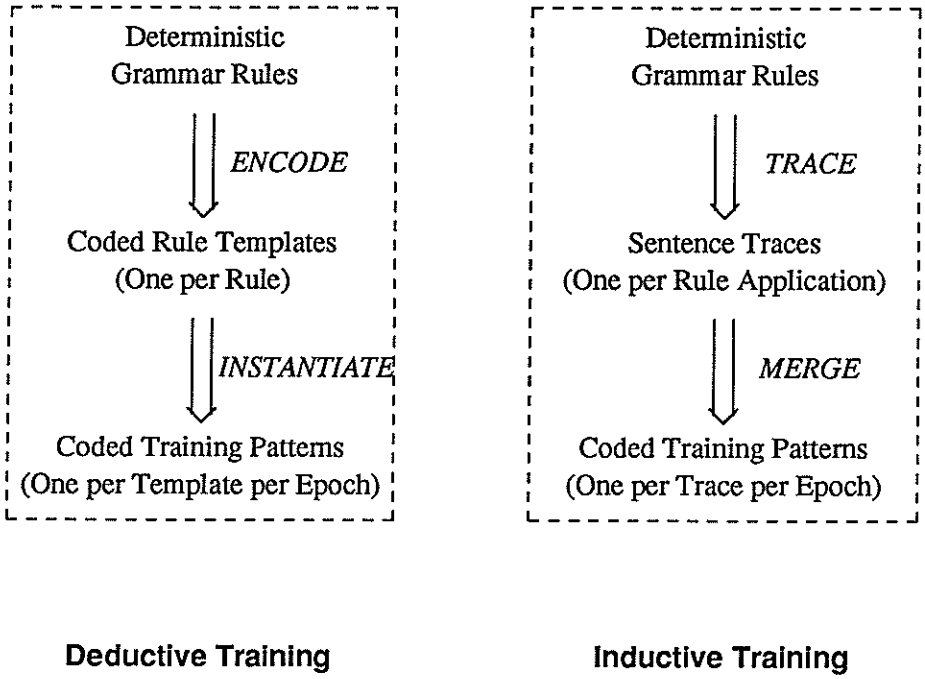

Figure 1:
Extraction of Deductive and Inductive Training Patterns

The task of the inductively-trained hybrid parser parallels that of LPARSIFAL [3]. In LPARSIFAL the object is to learn (symbolic) grammar rules from examples of correct sentences. The success of this task is gauged by directly comparing the rules learned to those of PARSIFAL. In the hybrid parser inductive training requires the network to exhibit the correct rule-following behavior after being trained with a sample of sentence traces. Training occurs through the mechanism of backward propagation. No symbolic rules are learned as such, but the behavioral characteristics of the rules are captured within the parameters of the network.

The hybrid parser, therefore, can exhibit different properties depending on the patterns used in training. Inductive learning is a much more tedious process since much more data is required as compared to that required for deductive training. Also, the range of sentence types handled depends greatly on the completeness of the examples presented. Deductive training imposes an ordering on the training patterns that assures a completeness which is difficult to achieve with inductive training, but inductive training patterns reflect the frequency of rule occurrences seen in actual sentence processing.

A third method of training combines both deductive and inductive patterns. This "mixed" training promises to be superior to either of two alone. Networks trained in this way exhibit good generalization characteristics, as expected from deductive training, as well as excellent performance on specific sentences, as expected from inductive training.

2.3. Neural Networks and Backward Error Propagation

In the burgeoning field of Connectionism, a technique known as Backward Error Propagation [47,56] has gained increasing acceptance. This training technique is based on gradient descent in the space of an error function.[1] It is ordinarily seen in networks containing multiple layers with the initial layer designated as the input layer and the final layer designated as the output layer. Further background on this and related techniques can be found in [23].

We have already shown a significant improvement in the training times for our networks. The improvements [28] utilize a better error function in combination with the conjugate gradient method for training [45]. The improved error function significantly reduces the problem of unwanted saddle points. This problem is especially acute for large networks such as the ones we have encountered. As a general rule, we find that the most effective techniques for speeding up learning for large networks involve studying more global properties of the network such as the weights (connections) in preference to examining computational properties of the unit.

Both steepest descent (of which "backprop" is a version despite all protestations) and conjugate gradient are only guaranteed to find sets of weights which make the gradient zero. This includes all local and global maxima and minima as well as saddle points. "Backprop" uses several heuristics to avoid local zeros of the gradient but it cannot guarantee to miss them all. Conjugate gradient is a Newton (second derivative) method and it converges superlinearly, perhaps quadratically. The gradient descent method converges linearly. This means that the distance from the gradient to zero goes as the first power of the distance of the previous iteration for gradient descent and a power greater than the first for conjugate gradient. This is usually very desirable. However if a global minimum is desired the conjugate gradient method will tend to find local extrema and saddle points at a much higher rate than the "backprop" method. The error function described in [28] removes most if not all of this objectionable behavior for the back propagation training of feed forward neural networks.

Convergence is an asymptotic concept. It is possible that a low order method will occasionally outperform a high order method; especially if not too close a convergence is acceptable. But over time the higher order method will be superior.

Recently, a particular variety of network structure which contains feedback connections has been investigated by Pollack [44] to perform various stack and tree-like operations. These recursive auto-associative memory (RAAM) networks compactly represent such structures and support functional composition of their operations. Although his results are still somewhat limited, they show much promise that this technique can be developed into a viable method for representing standard data structures as distributed patterns.

We have done some preliminary investigations on this technique. These are reported in [28] and show that distributed representations can indeed be manipulated as a stack. We are continuing these investigations as part of our research.

3. Studies

As stated earlier, the goals of this research are to examine the marriage of symbolic, deterministic parsing with sub-symbolic, connectionist processing and to investigate the advantages and disadvantages of various architectures. These goals have been partially met in the sense that we have already examined

[1] A complaint: Back propagation is the technique by which the gradient is computed not the technique by which the global minimum is found. Most neural network texts use the term "backpropagation" for the latter. A more appropriate appellation might be "heuristic gradient descent."

several networks and grammars for syntactic processing. We are continuing this work by examining several more. Furthermore, our aim is to determine how well the narrowly-defined goals of syntactic processing can be expanded to even wider contexts. In this section, a plan for accomplishing these goals is discussed.

In the first section, we describe basic research on hybrid systems as a natural continuation of our recent work. In recognition of the need to factor in other constraints while parsing, we next describe research on lexical access which is compatible with our hybrid parser. Next, we describe a new architecture for deterministic parsing which moves away from heavy dependence on the architecture of PARSIFAL and closer to distributed representations and connectionism. Finally, since the research depends heavily on neural network training methods we describe research aimed at supporting our particular demands through innovative and faster training methods.

The work on distributed representations is seen as replacing the architecture we now have once it can be established that the new architecture is superior. All other tasks discussed in this report are complementary and thus can be combined with each other to enhance the overall system.

3.1. Hybrid System Research

As mentioned earlier, we have enjoyed some success in building a syntactic parser, composed of symbolic and sub-symbolic parts, for processing inputs which are not always well-formed according to the grammar rules. The research described in this section continues those explorations along several lines.

Although we have developed several distinct grammars, we make no claims about the completeness of any of them. Figure 2 summarizes those grammars. The large grammar is comparable to the most complete deterministic grammars published and is based on the largest grammar given by Marcus [34]. Grammar construction and development can be a tedious process. In fact, some researchers devote years of their professional lives to such activities. Our goal is to produce grammars similar to the ones already done, but aimed at demonstrating specific features of our work.

In the large grammar, we chose to use only two buffer positions. We know how this affects symbolic deterministic parsers, but can only conjecture about hybrid parsers. We believe going to three buffer positions, while increasing the network size and training times, will enhance its generalization capabilities and create fewer local ambiguities. We have begun to examine these effects by contrasting several medium size grammars, some which contain two buffer positions and some which contain three.

In the figure, we show the number of presentations necessary to train each grammar and, where we have conducted the experiments, the data for both gradient descent (backpropagation) and conjugate gradient. These data should be considered upper bounds on the amount of actual training necessary. In many cases, convergence to an acceptable set of weights happens more quickly than the number of presentations shown. As discussed earlier, a convergence criteria is not well-defined, the point of acceptability is difficult to measure. Thus, training involves estimating an upper bound on the number of presentations required and presenting patterns to the network that many times.

We have experience in performing both deductive and inductive training with our grammars. We lack experience in combining deductive with inductive into mixed training sequences, although preliminary data shows this to potentially be the most powerful technique. We are continuing to experiment with these techniques in combination.

How input items are coded depends on the way in which the grammar is written. We have chosen encodings based primarily on syntactic features (e.g., noun, prep, aux-verb-attached, etc.). Others [38] have chosen to use semantic microfeatures. Clearly the choice of features limits the range of information

	Target Grammars				
	Small	Medium			Large
Number of Rules	13	22	24	24	73
Number of Actions	5	20	22	22	40
Network Size:					
Units	44-15-5	35-20-20	37-25-22	52-30-22	66-40-40
Weights	755	1140	1272	2272	4320
Presentations (× K):					
Gradient Descent	150–350	500	600	600–2000	1000
Conjugate Gradient	15–25			35–60	
Based On Reference	Example (Winston, 1984)	Appendix C (Marcus, 1980)			Appendix D (Marcus, 1980)

Figure 2:
Summary of Target Grammars Used in Hybrid Parser

available about each item. We are experimenting with several schemes for this encodement. As a particular case, what can be done with novel or unknown words (i.e., words not in the lexicon) based on their syntactic context? For example, a person's name is not likely to be in a lexicon, unless it is ambiguous as a common English word (e.g., bill, bob, carol). If the parser performs analogous to how it processes a lexically ambiguous item, we should find some ability for the parser to treat the item as appropriate to its context. Further, we are investigating a method for back-propagating error values to unknown items so that lexical entries can be developed for unknow items and they can be transformed in time to their proper entry. This method of representation is based on [40].

We are systematically evaluating our hybrid system by expanding our corpus of sentence forms to range across a broader spectrum of sentence structures. Each of these is being tested against an appropriate grammar in order to establish the extent and boundaries of the techniques.

3.2. Lexical Access Research

Developing a robust lexical analyzer should aid in testing our hypothesis of mutually constraining components. Activation patterns that emerge from the lexicon should interact with syntactic processing to produce reasonable interpretations of novel and ambiguous words.

The core of the current system is the syntactic parser, and this will provide the main focus of the research. However, we believe that what goes on *between* the components of a system is at least as important as what goes on *within* a component. For this reason, we are extending the current hybrid system by including lexical and some semantic processing.

Having decided to include other knowledge sources, what sort of architecture do we envisage? Debate continues on the question of whether there should be what Crain and Steedman [11] have called a 'strong interaction' between components, where a higher level component is able to direct the processing of a lower-level one, or whether the interaction should be 'weak', the higher level only selecting among various candidates provided by the the lower level. Even if we allow only a weak interaction, further questions remain: should a higher level be able to fill in missing information, or correct what appears to be erroneous information? how could this be done without opening the door to the kind of hallucinatory processing of earlier speech recognition machines?

Connectionist architectures seem to be particularly suited to exploring these sorts of questions. Work by McClelland and Rumelhart [39], McClelland and Elman [37], Cottrell [9], among others, has shown how different parts of a connectionist system might "seep" into each other like the hues of a water color. This meshing of components is facilitated by the fact that connectionist systems are all made of the same stuff. The uniform networks of nodes and lattices provide a basis for the merging of different kinds of information. The flexible play between excitation and inhibition, activity and feedback, allows much more delicate interaction between components than is usually possible in a more traditional system. Constraint relaxation (e.g., allowing for mispronounced words, or ungrammatical sentences) has to be provided explicitly in a symbolic system. Usually this kind of exception handling must be applied indiscriminately because it is difficult to say, at the level it is applied, whether it is in fact necessary.

Connectionist systems are ideally suited to the mobilizing of mutual constraints, and, especially in winner-take-all networks, only show constraint relaxation in appropriate circumstances. The tighter the constraint provided by the context, the greater the excitations and feedback at the higher level. For example, in a system such as TRACE [37], an ambiguous stop segment at the beginning of /ip/ would be less likely to be identified as /d/ then if it were at the beginning of /im/. The excitation of /ip/ would be shared out between tip, dip, pip, and kip, whereas in the latter case, dim would receive most, if not all, of the activation. This sensitivity to the overall state of competition is extremely hard to emulate in a symbolic system.

The input to the current hybrid parsing system is managed by a symbolic component which places into the buffer the kinds of syntactic category information that a simple lexicon would provide. The output from the parser is a sequence of coded actions from which one can construct a parse tree for the sentence. It would be a fairly simple matter to increase the sophistication of the input processing. Connecting the parser to a component for semantic interpretation would be a more difficult task.

We are examining Cottrell's substantial work on word sense disambiguation and connectionist parsing. However, our focus is slightly different. Whereas he built a parser because he needed one for his lexical work, we are building a lexicon because we need one for our parsing work. Cottrell's first concern, quite rightly, was to see if such phenomena as semantic priming could be modeled within a connectionist lexical access system. He left much of the work of combining components for future work. Our main concern is to examine the interactions between components. We feel our work should dovetail well with Cottrell's.

First of all, we are extending the symbolic part of the system, adding lexical and semantic components. We expect to pursue this by incorporating the work on word experts [1,25,50]. As stated previously, we are particularly interested in the interactions between components. The hybrid system is providing a starting point for such work.

There are two aspects of Cottrell's work that we are pursuing further. The first is the interlinking of the connectionist components. Other connectionist researchers have worked "down" the speech chain (e.g., McClelland and Elman's work [37] linking phonetic, phonemic, and lexical levels.) Following

Gigley's HOPE system [19,20], we are interested in working "up" the chain, linking lexical, syntactic, and semantic levels. using the R^2AAM mechanisms described below.

3.3. Distributed Representation Research

At the heart of the architecture of a deterministic parser is the use of a fixed-length buffer. Input items flow into the buffer and ultimately become part of a structure when attached from the end of the buffer. All other features of deterministic parsing — rules, rule packets, priorities, attention-shifting, actions, and the grammar itself — are arbitrary artifacts and could be realized quite differently if desired. In fact, it has been shown [49] that the stack and buffer can be combined to simplify processing. The buffer is the interface between the parser and the perceptual modality. This leads to an important question: can the arbitrary components of deterministic parsing, that is everything except the buffer, be realized as a connectionist network and would there be advantages in constructing such a system?

Let us first consider the advantages. A weakness in the hybrid parser is the requirement that a set of grammar rules exist for deductive training or that a large set of examples exist for inductive training. Ideally, one would like to train the system extensionally with examples of inputs and features of their results. This approach is consistent with what has been called "extensional programming."

Another weakness is the use of symbolic actions to build symbolic structure. Just as in any symbolic program, one small error creates a different program, often one with a bug. This leaves our hybrid system with some of the same weaknesses as symbolic parsers. To gain robustness, the creation of structure must not be performed by a sequence of actions. Rather, given the monotonicity of structure in a deterministic parser, one structure can be mapped to a slightly larger structure. This would result in a much more robust system. Such a mapping should be possible using distributed representations provided by connectionism. Part of aim in this research is to find such mappings.

To be successful, this plan requires that the network itself be capable of representing both stacks and trees, since these structures would no longer be maintained symbolically. It must be capable of representing a stack because stacks are essential to processing embedded structures. It must be capable of representing a tree if indeed the process of parsing is to produce a parse tree or hierarchical structure. At first, such requirements seem difficult to meet without severely restricting the parser produced. Indeed, as mentioned earlier, others who have attempted to build connectionist parsers have accepted some rather severe limitations such as bounded sentence length or the requirement that an entire sentence be presented at one time as input. But these restrictions contradict the iterative nature of the process.

Pollack [44] describes a method for implementing arbitrary stacks and trees in a recursive auto-associative memory (RAAM). The idea is to provide neural network circuitry to perform the basic stack operations of push and pop. The pop operation necessarily guarantees to extract the last element pushed onto the stack. The push operation guarantees to compress the stack and new element into a stack that pop can operate upon. Similarly, tree structures can be assembled or decomposed. As an important feature, these structures can be composed arbitrarily and hence are compositional, at least in the functional sense [18].

Deterministic parsing requires a stack whose elements are trees. This leads us to a further generalization of the RAAM idea. We propose a Recursive Recursive Auto-Associative Memory (R^2AAM) which permits the composition of elements which are themselves tree RAAMs into a stack. Such a memory is recursive in its elements as well as in how the elements combine. At this point in our research, no experiments have been performed to confirm how well these memory structures will perform. However, the extension of Pollack's ideas to what we require seems very direct.

Remaining is the question of how to know if proper structures are being built in the R^2AAM. For this, we propose to use an idea from St. John and McClelland [52] involving probe units. A collection of these units are included such that when a given probe is activated, an encoding of the corresponding piece of structure should appear in the output. It can then be observed if proper encodements are being constructed. Note that the term "structure" here does not necessarily mean syntactic structure and could conceivably be any identifiable activation pattern, perhaps one that would require RAAM decomposition.

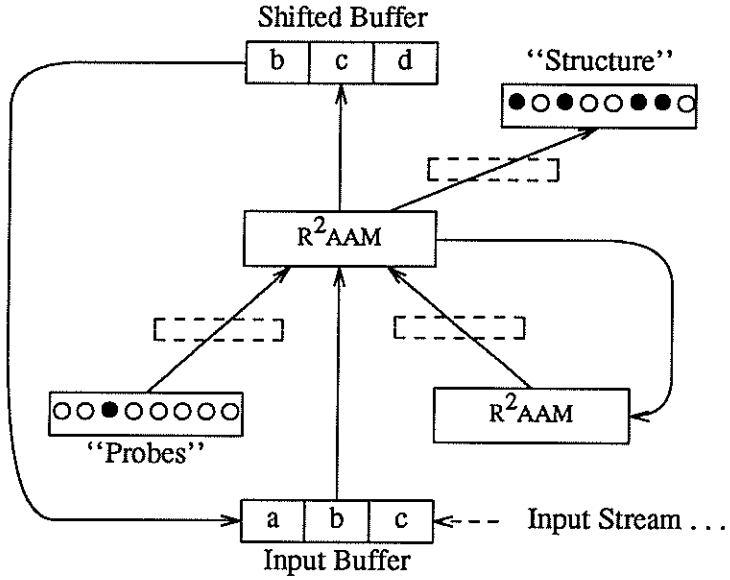

Figure 3:
Toward Distributed Representations in Deterministic Parsing

Now we can address the possibility of building a deterministic parser which uses distributed representations with probes to elicit the structural components of the result and with no rules or actions per se to provide an additional source of weakness. Figure 3 shows the design. The input stream fills the input buffer as constituents are processed. The primary task of the network is to shift the input buffer left by one element while internalizing the first element. The element shifted off the end must be represented in the R^2AAM in an appropriate and final way. Note that if processing follows the steps of PARSIFAL, this could require some extremely complicated processing, including the dropping and attaching of several structures. If this proves to be problematic, several small variations are possible to permit "worst case" scenarios.

Tests can be conducted with probes to verify that the element has indeed been properly represented. In fact, the probes must be integrated into the training. The shifted buffer is copied back to the input buffer, with the input stream filling in the missing element. In fact, due to the timing constraints of real-time language processing, the input stream might not have the next token ready for processing requiring a wait for the next token. The R^2AAM from the previous iteration becomes part of the new input. The dashed boxes represent possible additional layers of network. When an empty buffer occurs, parsing is finished and the final structure is accessible through the probes.

This design is an evolution which still subscribes to the determinism hypothesis, but does not demand a complete set of grammar rules for training. Instead, training is based on the resultant structures we require and the step-by-step process for getting there. The shifting operation simulates the need to decide how to process each input item. From the determinism hypothesis, we know that each input element at the left end of the buffer must be processed into at least a partial structure. Of course, the choice of three buffer positions is also based on assumptions made in deterministic parsing.

We are in the process of constructing a parser based on these ideas and evaluating its properties. We believe that it provides significant advantages over previous work. Part of our effort is in developing and refining the RAAM technique into a technique for R^2AAM. We are also experimenting with representations for probe outputs. Since complex structures will need to be represented, the most flexible representation may be one also based on RAAMs. Finally, we are developing training data for a range of sentences.

3.4. Neural Network Research

This part of the research is driven by problems generated in the other parts. Our present needs involve experimentation with recursive and recurrent networks of the type discussed above, techniques for training with rule templates, and improvements to current methods.

Even for networks of moderate size, training can take an enormous amount of processing time. The choice of training strategy will have an impact on this. There are two types of methods for improving the speed with which we can train networks:

(1) Methods involving improvements to the mechanism of training. For example, the conjugate gradient method [45], quickprop [14], cascade correlation [15], or other methods can be used.

(2) Methods involving changes to the presentation schedule of the patterns to be learned. For deductive training, the probability that a "don't care" symbol (?) becomes a +1 or -1 can be manipulated so that learning takes place in stages.

Too much emphasis is currently being placed on methods of the first type. Our experiments [28] show the second type of training to also be an important technique for improving convergence speeds. Additionally, the quality of the resulting network can be greatly improved. These and other strategies are being investigated in concert with problems generated by the other tasks in this research.

3.5. Evaluation

Evaluation of this work is made by selecting examples from published work on deterministic parsing and comparing our results to those. In order to know the effectiveness of our hybrid architectures, we need to have such a basis for comparison. Fortunately, there is a rich tradition of work in this area [2-4,6,7,24,33,34,41,42,46,49] from which we can draw examples. Where examples do not exist, we are evaluating our methods according to the structures produced and the degree to which, in our judgement, they capture the structure of the sentence form.

4. Impact of Research

Previous attempts to introduce connectionism in processing natural language have fallen subject to criticism for either permitting only sentences of limited length or for demonstrating only very limited language features in small-scale grammars. We have an approach which overcomes the first limitation and have demonstrated it using a relatively mature grammar. We recognize the need to continue the scaling-up process to evaluate and test how well the architecture of our hybrid system can be expanded to include more of the language processing task.

Our approach to NLP is distinct in several respects. We have elected to take a hybrid approach which allows us to draw on both the research tradition of symbolic approaches as well as the robustness and generalization capabilities that neural networks bring. The strengths of the two approaches complement each other well for deterministic parsing. A goal of this research is to further demonstrate the potential for symbiosis between the two approaches.

In scaling up from a small toy grammar to larger and larger grammars, it has actually gotten easier to demonstrate robustness and generalization in the processing of language. As we continue to expand our system by investigating the constraints provided by other processing components, we expect to see the critical constraints of language processing increasingly coming into play, much like Waltz [54] discovered in his seminal work on vision. While Birnbaum's [5] arguments against autonomous syntactic processing may have the ring of truth, we believe that syntactic processing should not be performed in a vacuum and further that important and significant results can be attained, even while limiting processing to syntax, when ambiguity is built into the representation itself as a distributed pattern. These patterns in turn should be amenable to influences from other components in a traditional language processing environment.

The impact on attempts to handle language forms that are ungrammatical or lexically ambiguous should also be apparent. Symbolic approaches to processing ill-formed inputs [7,32,55] have enjoyed only limited success. We have compared our performance with that of other published work and it compares very favorably. Our efforts in this research are being compared with a variety of systems from the literature in terms of the ability to correctly process a broad range of sentence forms. We aim to fully characterize the value of the techniques described here.

Finally, in the course of this research, we are developing perspectives on how rule-based systems generally can benefit from the introduction of connectionism. We have argued [16,30,31] that rules can serve as the basis for deductive training sequences. This type of training has a place in knowledge acquisition in which an expert or a textbook articulates general rules. Inductive training is analogous to experiential training which keys on actual processing cases.

We are continuing to draw general lessons from our specific domain of language processing.

References

1. Adriaens, G. and S. Small, "Word Expert Parsing Revisited in a Cognitive Science Perspective," in *Lexical Ambiguity Resolution: Perspectives from Psycholinguistics, Neuropsychology, and Artificial Intelligence*, ed. S. Small, G. Cottrell, and M. Tanenhaus, Morgan Kaufman, San Mateo, CA, 1988.
2. Allen, James, *Natural Language Understanding*, Benjamin/Cummings, Menlo Park, CA, 1987.
3. Berwick, Robert C., *The Acquisition of Syntactic Knowledge*, MIT Press, Cambridge, MA, 1985.
4. Berwick, Robert C., "Locality Principles and the Acquisition of Syntactic Knowledge," PhD Thesis, MIT , Cambridge, MA, 1982.
5. Birnbaum, Lawrence L., "A Critical Look at the Foundations of Autonomous Syntactic Analysis," in *Proceedings of the 11th Annual Conference of The Cognitive Science Society*, pp. 99-106, Lawrence Erlbaum Associates, Hillsdale, NJ, August 16-19, 1989.
6. Briscoe, E.J., "Determinism and its implementation in PARSIFAL," in *Automatic Natural Language Parsing*, ed. K. Sparck Jones and Y. Wilks, pp. 61-68, Ellis Horwood, Chichester, England, 1983.
7. Charniak, Eugene, "A Parser with Something for Everyone," in *Parsing Natural Language*, ed. M. King, pp. 117-150, Academic Press, New York, NY, 1983.
8. Cole, R. A. and J. Jakimik, "A Model of Speech Perception," in *Perception and Production of Fluent Speech*, ed. R. Cole, Lawrence Erlbaum Associates, Hillsdale, NJ, 1980.
9. Cottrell, Garrison W., *A Connectionist Approach to Word Sense Disambiguation*, Pitman Publishing, London, 1989.
10. Cottrell, Garrison W. and Fu-Sheng Tsung, "Learning Simple Arithmetic Procedures," in *Proceedings of the 11th Annual Conference of The Cognitive Science Society*, pp. 58-65, Lawrence Erlbaum Associates, Hillsdale, NJ, August 16-19, 1989.
11. Crain, S. and Steedman, "The Use of Context by the Psychological Parser," Paper presented at the Symposium on Modeling Human Parsing Strategies, Center for Cognitive Science, University of Texas at Austin, 1981.
12. Elman, Jeffrey L., "Structured Representations and Connectionist Models," in *Proceedings of the 11th Annual Conference of The Cognitive Science Society*, pp. 17-25, Lawrence Erlbaum Associates, Hillsdale, NJ, August 16-19, 1989.
13. Engelmore, Robert and Tony Morgan, Eds., *Blackboard Systems*, Addison-Wesley, Reading, MA, 1988.
14. Fahlman, Scott E., "Faster Learning Variations on Back-Propagation: An Empirical Study," in *Proceedings of the 1988 Connectionist Models Summer School*, Morgan Kaufmann, San Mateo, CA, 1988.
15. Fahlman, Scott E. and Christian Lebiere, "The Cascade-Correlation Learning Architecture," Technical Report CMU-CS-90-100, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, February 14, 1990.
16. Faisal, Kanaan A. and Stan C. Kwasny, "Deductive and Inductive Learning in a Connectionist Deterministic Parser," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2, pp. 471-474, Washington, DC, January 15-19, 1990.
17. Fanty, Mark, "Context-Free Parsing in Connectionist Networks," Technical Report 174, Computer Science Department, University of Rochester, Rochester, NY, 1985.

18. Gelder, Timothy van, "Compositionality and the Explanation of Cognitive Processes," in *Proceedings of the 11th Annual Conference of The Cognitive Science Society*, pp. 34-41, Lawrence Erlbaum Associates, Hillsdale, NJ, August 16-19, 1989.

19. Gigley, H., "Grammar Viewed as a Functioning Part of a Cognitive System," in *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, University of Chicago, Chicago, IL, July 8-12, 1985.

20. Gigley, H., "Process Synchronization, Lexical Ambiguity Resolution, and Aphasia," in *Lexical Ambiguity Resolution: Perspectives from Psycholinguistics, Neuropsychology, and Artificial Intelligence*, ed. S. Small, G. Cottrell, and M. Tanenhaus, pp. 229-267, Morgan Kauffman, San Mateo, CA, 1988.

21. Grosjean, F., "Spoken Word Recognition Processes and the Gating Paradigm," *Perception and Psychophysics*, vol. 24, pp. 267-283, 1980.

22. Harrington, J. M. and A. M. Johnstone, "The Effects of Equivalence Classes on Parsing Phonemes into Words in Continuous Speech Recognition," *Computer Speech and Language*, vol. 22, pp. 273-288, 1987.

23. Hecht-Nielsen, Robert, *Neurocomputing*, Addison-Wesley, Reading, Ma, 1990.

24. Hindle, Donald, "Deterministic Parsing of Syntactic Non-Fluencies," in *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pp. 123–128, 1983.

25. Hirst, Graeme, *Semantic Interpretation and the Resolution of Ambiguity*, Cambridge University Press, Cambridge, England, 1987.

26. Johnstone, A. M., "The Effects of Word Boundary Ambiguity on Lexical Access in Automatic Continuous Speech Recognition," PhD Thesis, Edinburgh University, Edinburgh, Scotland, 1989.

27. Jordan, Michael, "Serial Order: A Parallel Distributed Processing Approach," Technical Report 8604, Institute for Cognitive Science, University of California, San Diego, La Jolla, CA, 1986.

28. Kalman, Barry L., "Super Linear Learning in Back Propagation Neural Nets," Technical Report WUCS-90-21, Department of Computer Science, Washington University, St. Louis, MO, February, 1990.

29. Kwasny, Stan C. and Kanaan A. Faisal, "Connectionism and Determinism in a Syntactic Parser," *Connection Science: Journal of Neural Computing, Artificial Intelligence, and Cognitive Research: Special Issue on Connectionist Research on Natural Language*, vol. 2, no. 1–2, pp. 63–82, Carfax Publishing Company, Abingdon, Oxfordshire, England, 1990.

30. Kwasny, Stan C. and Kanaan A. Faisal, "Overcoming Limitations of Rule-Based Systems: An Example of a Hybrid Deterministic Parser," in *Konnektionismus in Artificial Intelligence und Kognitionsforschung*, ed. Georg Dorffner, Informatik-Fachberichte, Springer-Verlag, September, 1990. Austrian Society for Artificial Intelligence (OeGAI), Salzburg, Austria

31. Kwasny, Stan C. and Kanaan A. Faisal, "Rule-Based Training of Neural Networks," *Expert Systems with Applications: Special Issue on Applying Artificial Neural Networks to Expert Systems*, Pergamon Press, 1990. (in press)

32. Kwasny, Stan C. and Norman K. Sondheimer, "Relaxation Techniques for Parsing Ill-Formed Input," *American Journal of Computational Linguistics*, vol. 7, no. 2, pp. 99-108, 1981.

33. Marcus, Mitchell P., "Diagnosis as a Notion of Grammar," in *Advance Papers of the Workshop on Theoretical Issues in Natural Language*, ed. Roger C. Schank and Bonnie Nash-Webber, Cambridge, MA, 1975.

34. Marcus, Mitchell P., *A Theory of Syntactic Recognition for Natural Language*, MIT Press, Cambridge, MA, 1980.

35. Marslen-Wilson, W. D., *Parallel Processing in Spoken Word Recognition*, 1986. Manuscript

36. Marslen-Wilson, W. D. and L. K. Tyler, "The Temporal Structure of Spoken Language Understanding," *Cognition*, vol. 8, pp. 1-71, 1980.

37. McClelland, James L. and Jeffrey L. Elman, "The TRACE Model of Speech Perception," *Cognitive Psychology*, vol. 18, pp. 1-86, 1986.

38. McClelland, James L. and Alan H. Kawamoto, "Mechanisms of Sentence Processing: Assigning Roles to Constituents of Sentences," in *Parallel Distributed Processing*, ed. David E. Rumelhart and James L. McClelland, pp. 272-325, MIT Press, Cambridge, MA, 1986.

39. McClelland, James L. and David E. Rumelhart, "An Interactive Activation Model of Context Effects in Letter Perception; Part 1: An Account of Basic Findings," *Psychological Review*, vol. 88, pp. 375-407, 1981.

40. Miikkulainen, R. and M. G. Dyer, "Forming Global Representations with Extended Backpropagation," in *Proceedings of the IEEE Second Annual Conference of Neural Networks (ICNN-88)*, San Diego, CA, 1988.

41. Milne, Robert, "Resolving Lexical Ambiguity in a Deterministic Parser," *Computational Linguistics*, vol. 12, no. 1, pp. 1-12, January-March, 1986.

42. Nozohoor-Farshi, R., "Context-Freeness of the Language Accepted by Marcus' Parser," in *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, pp. 117-122, 1987.

43. Pinker, Steven and Alan Prince, "On Language and Connectionism: Analysis of a Parallel Distributed Processing Model of Language Acquisition," in *Connections and Symbols*, ed. Steven Pinker and Jacques Mehler, pp. 73-193, MIT Press, Cambridge, MA, 1988.

44. Pollack, Jordan B., "Recursive Distributed Representations," Report 89-JP-RECURSIVE,, Laboratory for Artificial Intelligence Research, The Ohio State University, Columbus, Ohio, August, 1989.

45. Press, William H., Brian P. Flannery, Saul A. Teukolsky, and William T. Vettering, *Numerical Recipes in C*, Cambridge University Press, Cambridge, England, 1988.

46. Ritchie, Graeme D., "The implementation of a PIDGIN interpreter," in *Automatic Natural Language Parsing*, ed. K. Sparck Jones and Y. Wilks, pp. 69–80, Ellis Horwood, Chichester, England, 1983.

47. Rumelhart, David E., Geoffrey Hinton, and Ronald J. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing*, ed. D.E. Rumelhart and J.L. McClelland, pp. 318-364, MIT Press, Cambridge, MA, 1986.

48. Selman, B. and G. Hirst, "A Rule-Based Connectionist Parsing System," in *Proceedings of the 7th Annual Conference of The Cognitive Science Society*, pp. 212-221, Lawrence Erlbaum Associates, Hillsdale, NJ, 1985.

49. Shipman, David W. and Mitchell P. Marcus, "Towards Minimal Data Structures for Deterministic Parsing," in *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pp. 815-817, Tokyo, Japan, August, 1979.

50. Small, S. and C. Rieger, "Parsing & Comprehending with Word Experts," in *Strategies for Natural Language Processing*, ed. W. Lehnert and M. Ringle, Lawrence Erlbaum Associates, Hillsdale, NJ, 1982.

51. Sondheimer, Norman K., "The Rate of Progress in Natural Language Processing," in *Theoretical Issues in Natural Language Processing*, ed. Yorick Wilks, pp. 113-117, Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

52. St.John, M.F. and J.L. McClelland, "Learning and Applying Contextual Constraints in Sentence Comprehension," Technical Report AIP-39, Department of Psychology, Carnegie-Mellon University, Pittsburgh, PA, June 8, 1988.

53. Waltz, David L. and Jordan B. Pollack, "Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation," *Cognitive Science*, vol. 9, pp. 51-74, 1985.

54. Waltz, David L., "Understanding Line Drawings of Scenes with Shadows," in *The Psychology of Computer Vision*, ed. Patrick H. Winston, McGraw-Hill, New York, NY, 1975.

55. Weischedel, Ralph M. and Norman K. Sondheimer, "Meta-Rules as a Basis for Processing Ill-Formed Input," *American Journal of Computational Linguistics*, vol. 9, no. 3-4, pp. 161-177, 1983.

56. Werbos, Paul, "Beyond Regression: New Tools for Prediction and Analysis in Behavioral Science," PhD Thesis, Harvard University, Cambridge, Ma, 1974.

57. Wilks, Yorick Ed., *Theoretical Issues in Natural Language Processing*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1989.

58. Winograd, Terry and F. Flores, *Understanding Computers and Cognition*, Addison-Wesley, Reading, Ma, 1987.